A discontinuous Galerkin fast-sweeping Eikonal solver for fast and accurate traveltime computation in 3D tilted anisotropic media

Philippe Le Bouteiller, Mondher Benjemaa, Ludovic Métivier, Jean Virieux

A discontinuous Galerkin fast-sweeping Eikonal solver for fast and accurate traveltime computation in 3D tilted anisotropic media

Philippe Le Bouteiller*, Mondher Benjemaa†, Ludovic Métivier*‡, and Jean Virieux*

* *ISTerre, Univ. Grenoble Alpes, F-38000 Grenoble, France. E-mail: philippe.le-bouteiller@univ-grenoble-alpes.fr*

† *Laboratory of Stability and Control of Systems and Nonlinear PDE, University of Sfax, 3029 Sfax, Tunisia*

‡ *LJK, Univ. Grenoble Alpes, CNRS, F-38000, Grenoble, France*

(December 10, 2018)

Running head: **Fast Sweeping DG for Eikonal equation**

ABSTRACT

We tackle the challenging problem of efficient and accurate seismic traveltime computation in 3D anisotropic media, by applying the fast sweeping method to a discontinuous Galerkin-based Eikonal solver. Using this method leads to a stable and highly accurate scheme, which is faster than finite-difference schemes for given precision, and with a low computational cost compared to the standard Runge–Kutta discontinuous Galerkin formulation. The integral formulation of the discontinuous Galerkin method also makes it easy to handle seismic anisotropy and complex topographies. Several numerical tests on complex models, such as the 3D SEAM model, are given as illustration, highlighting the efficiency and the accuracy

of this new approach. In the near future, these results will be used together with accurate solvers for seismic amplitude and take-off angle computation in order to revisit asymptotic inversion (traveltime/slope tomography) and imaging approaches (quantitative migration involving amplitudes and angles).

INTRODUCTION

Asymptoptic approaches based on traveltime and amplitude computation in the high frequency approximation are widely used in many seismic applications such as traveltime/slope tomography for initial velocity model building (Le Meur, 1994; Hole and Zelt, 1995; Billette and Lambaré, 1998; Leung and Qian, 2006; Taillandier et al., 2009; Lelièvre et al., 2011; Tavakoli F. et al., 2017), or ray-based (quantitative) migration (Beylkin, 1985; Bleistein, 1987; Beylkin and Burridge, 1990; Jin et al., 1992; Gray and May, 1994; Lambaré et al., 2003; Operto et al., 2003). The advantage of a slowly varying quantity (traveltime) compared to the highly oscillating wavefield makes possible efficient decimations/interpolations over sparse grids for storage, which might be of great interest when working with large models and datasets (Mendes, 2000; Vanelle and Gajewski, 2002; Alkhalifah, 2011).

In the high-frequency regime, we may consider the Lagrangian framework of the ray theory to compute traveltimes by tracing the characteristics of the Eikonal equation (Červený, 2001). Solving the related ordinary differential equation with initial conditions (source location and shooting angle) is straightforward and easy to handle. However, when boundary conditions are considered (source and receiver locations), the two-point ray tracing can be a quite challenging task because of the non-uniform sampling of the medium by rays. Estimating traveltimes at a given point of the medium leads to sophisticated interpolation/extrapolation techniques, especially when considering shadow zones (Runborg, 2007). Also, when considering many source/receiver pairs, the ray tracing approach might become less efficient, as the computational cost scales as the product between the number of sources and receivers. Alternatively, when one is only interested in first-arrival traveltimes, solving the Eikonal equation within an Eulerian framework enables the computation of first-arrival

traveltimes everywhere in the medium, thanks to the concept of viscosity solution (Crandall and Lions, 1983, 1984): somehow, geometrical theory of diffraction is handled by the viscosity solution (Keller, 1962; Runborg, 2007). Moreover, the computational cost is only proportional to the number of sources, which makes it very efficient for dense acquisition settings. The drawback of the Eikonal approach is that it implies to solve a nonlinear partial differential equation (PDE) which requires sophisticated numerical tools to obtain reliable and accurate results.

Vidale (1988) promoted an expanding box framework, and triggered an abundant literature about Eikonal solvers. The original scheme of Vidale (1988) only computes traveltimes corresponding to outgoing rays. In heterogeneous media, the first arrival is thus not guaranteed, since some rays may go back into the expanding square, for instance in the presence of high velocity zones. Improvements of this technique are proposed in Podvin and Lecomte (1991), van Trier and Symes (1991), and Hole and Zelt (1995). Kim and Cook (1999) uses (Weighted) Essentially Non-Oscillatory (WENO) schemes (Shu and Osher, 1988, 1989; Jiang and Shu, 1996; Liu et al., 1994; Jiang and Peng, 2000) for local discretization, while choosing the expanding box framework ("Down'N'Out") for computing the global solution. A post-sweeping (PS) technique is added in order to retrieve the causality and thus first-arrival traveltimes, which leads to the ENO-DNO-PS algorithm. This is to some extent similar to the approach developed in Hole and Zelt (1995) based on additional reverse propagation steps. Instead of relying on an arbitrary squared-box expansion, wavefront-tracking schemes are proposed to better fit causality by following the expansion of the wavefront itself (Qin et al., 1992; Cao and Greenhalgh, 1994). Doing so, the post-sweeping technique is no longer necessary.

In their concepts, ENO-DNO-PS schemes are close to fast-sweeping methods (FSM)

and wavefront-tracking schemes are similar to fast-marching methods (FMM), two general classes of methods developed in the field of applied mathematics which have found applications in many domains in the recent years. These methods rely on an ordering of the nodes. FMM belongs to single-pass algorithms, based on Dijsktra's algorithm (Dijkstra, 1959), which considers the propagation front and makes it evolve (Tsitsiklis, 1995; Sethian, 1996, 1999). On the other hand, FSM belongs to multi-pass algorithms relying on global orderings of the nodes. All nodes are updated during each Gauss-Seidel iteration (sweep), following alternative orderings (Boué and Dupuis, 1999; Tsai et al., 2003; Zhao, 2005; Kao et al., 2004; Luo and Zhao, 2016). Both FMM and FSM have been intensively applied to solve the Eikonal equation in a wide range of problems. Extensive comparisons showing their numerical efficiencies can be found in Gremaud and Kuster (2006), and highlight that determining which strategy is the best is highly problem-dependent.

In presence of anisotropy, the Eikonal equation is more complex and needs adequate numerical strategies. The ENO-DNO-PS principles were extended to the anisotropic case in Dellinger and Symes (1997), Kim (1999), and Qian et al. (2001). Some extensions have also been carried out concerning FMM (Cristiani, 2009). They are based on approximations and they are generally difficult to implement. Mirebeau (2014) and Mirebeau and Portegies (2018) propose a new approach for the FMM for anisotropic Eikonal. However, the most mature strategies proposed so far in a geophysical context rely on FSM. The first extensions have focused on elliptical anisotropy (Tsai et al., 2003; Qian et al., 2007a), which could be handled quite naturally since it amounts to a dilation in space. The general 2D tilted transversely anisotropic (TTI) Eikonal comprises spatial derivatives of the traveltime to the power of four, which is more challenging. Han et al. (2017) propose to solve the related quartic equation and to select the appropriate root, yielding a high computational cost.

Tavakoli F. et al. (2015), Waheed et al. (2015), and Waheed and Alkhalifah (2017) prefer a fixed-point iteration technique to solve an elliptical equation at each iteration with a suitable right-hand side accounting for anellipticity.

Most of the FSM extensions to anisotropy considered 2D problems only, except in Waheed et al. (2015) where tilted orthorhombic (TOR) media are considered. Moreover, all the above-mentioned methods are developed using finite differences (FD) schemes, generally of first-order, or higher-order at high cost with non-compact stencils. The convergence order is in general less than one, or equal to one if the source point is handled correctly by using the *celerity domain*, the *factorization* or the *perturbation* methods for instance (Pica, 1997; Zhang et al., 2005a; Fomel et al., 2009; Luo and Qian, 2011; Noble et al., 2014). This results in Eikonal solvers that are efficient and simple to implement, but with a limited accuracy. Another strategy has been explored recently in Le Bouteiller et al. (2018), using a discontinuous Galerkin (DG) finite-element discretization instead of a FD approach, in order to increase the convergence order, and to obtain high accuracy on traveltimes and spatial derivatives in heterogeneous TTI media. Obtaining derived quantities like angles, amplitudes, or curvatures, with high accuracy, is crucial for tomographic/imaging methods. These quantities are based on first-order and second-order derivatives of the traveltime, so that second-order or third-order schemes are required for traveltime computation. Another advantage of such a finite-element approach is that the integral finite-element formulation can be performed in complex geometries, so that complex topographies can be precisely handled. This turns out to be of major interest when considering complex land targets (Improta et al., 2002; Taillandier et al., 2009).

Inspired by the work of Cheng and Shu (2007) and Cheng and Wang (2014), Le Bouteiller et al. (2018) consider a numerical scheme applicable to generic time-dependent Hamilton–

Jacobi equations, a class of equations to which Eikonal belongs, and make use of the Runge–Kutta time integration with suitable source condition until the steady state is reached, which corresponds to the sought traveltime solution. We shall refer to this method as the RK-DG solver. In terms of computational cost, the main bottleneck of this approach is the rather slow convergence in pseudo-time to reach the static solution through all transient states in every part of the medium, while the evolution of the front is localized. This results in an algorithmic complexity of $\mathcal{O}(\#\text{dof}^{3/2})$ where the total number of degrees of freedom is denoted by #dof.

It seems therefore natural to integrate the FSM approach in such formulation, with $\mathcal{O}(\text{N})$ complexity, as an acceleration tool for reaching the steady state. Such an idea is already tested in a FD framework by Zhang et al. (2005b), using FSM as an acceleration loop over the time-marching procedure. Regarding DG approaches, a numerical strategy is developed in 2D to solve Hamilton–Jacobi equations with a FSM procedure over a DG discretization by Li et al. (2008) and Zhang et al. (2011). Based on the DG solver of Cheng and Shu (2007), this strategy exhibits some practical limitations: an initial guess is needed and deduced from a preliminary FD computation; the local scheme requires a cumbersome least-squares L^2 reconstruction of the solution's derivatives at the cell interface. In addition, it is developed only for isotropic media, and the point source singularity is not cured.

We propose to integrate such FSM acceleration into the approach developed in Le Bouteiller et al. (2018). The implementation of FSM over this solver and the extension to three dimensions leads to an accurate and efficient solver for 3D traveltime computation, which we refer to as the FSM-DG solver. Based on the state-of-the-art DG scheme of Cheng and Wang (2014), the L^2 reconstruction of spatial derivatives is avoided. Without any need of initial solution guess, features such as point source singularity treatment, 2D-TTI and

3D-TOR anisotropy and complex topographies are integrated in this new approach yielding highly accurate traveltime estimation with a rather simple handling by potential users.

The remainder of the paper is organized as follows. First, the numerical method is detailed: the Eikonal equation and its causality-consistent DG discretization are recalled, an FSM strategy which consists of a local solver and a global Gauss-Seidel-based strategy is presented, and a mesh deformation strategy for topography handling is exhibited. Second, the accuracy and the efficiency of the resulting FSM-DG solver are illustrated through various examples in 2D and 3D: first in simple media for validation purpose, then in complex realistic media, such as the 3D SEAM model, with heterogeneities, topography, and anisotropy to illustrate the properties of the solver on challenging settings. In the last example, a comparison is performed with a full wavefield modeling. A conclusion closes this study.

NUMERICAL METHOD

Eikonal equation

Hereafter, we adopt the dynamic formulation of the Eikonal PDE in the Hamiltonian framework, which writes

$$\partial_\xi u(\mathbf{x},\xi) + \mathcal{H}(\mathbf{x}, \nabla_\mathbf{x} u(\mathbf{x},\xi)) = 0, \tag{1}$$

where the spatial coordinates $\mathbf{x}$ span the space $\mathbb{R}^d$ with $d = 2$ or $d = 3$, and ξ denotes a pseudo-time evolution parameter. In an isotropic medium, one can write

$$\mathcal{H}_{\mathrm{ISO}}(\mathbf{x}, \nabla_\mathbf{x} u(\mathbf{x},\xi)) = \|\nabla_\mathbf{x} u(\mathbf{x},\xi)\| - \frac{1}{c(\mathbf{x})}, \tag{2}$$

where the wave speed is denoted by $c(\mathbf{x})$. The stationary state of Hamilton–Jacobi equation 1 with the Hamiltonian of equation 2 verifies the static Eikonal equation $\mathcal{H} = 0$. At the stationary state, we have $\lim_{\xi \to \infty} u(\mathbf{x}, \xi) \equiv T(\mathbf{x})$ where $T(\mathbf{x})$ is the traveltime field. In a finite computational domain, we set the source boundary condition to $u(\mathbf{x_s}, \xi) = 0$ at any pseudo-time ξ at the source point $\mathbf{x_s}$, and such stationary state is obtained at a finite pseudo-time $\xi^\star$ once the source information have been propagated from the source to the entire domain. Using this time-marching procedure to reach the steady-state is studied in Zhang et al. (2005b), while a formal link between the static and the dynamic Hamilton–Jacobi equations is proposed by Osher (1993) through the level-set framework.

Following Le Bouteiller et al. (2018), we write the Hamiltonian for the 2D vertical transversely isotropic (VTI) case as

$$\mathcal{H}_{VTI} = d(u_{,x})^2 + e(u_{,z})^2 + c(u_{,x})^2(u_{,z})^2 - 1, \tag{3}$$

where the derivatives of $u(x, z, \xi)$ with respect to x and z are respectively denoted by $u_{,x}$ and $u_{,z}$, and with

$$\begin{cases} c = -2(\epsilon - \delta)V_P^4, \\ d = (1 + 2\epsilon)V_P^2, \\ e = V_P^2, \end{cases} \tag{4}$$

where the Thomsen's parameters are denoted by ϵ and δ (Thomsen, 1986), and the P-wave velocity along the vertical axis is denoted by V_P. This derivation comes from Christoffel's dispersion relation in an elastic medium (see e.g. Červený, 2001; Slawinski, 2003), considering only the coupled P–SV propagation mode, under the acoustic approximation (Alkhalifah, 2000). The tilted (TTI) case is retrieved by applying the local rotation by the

angle $\theta(\mathbf{x})$ between the local rotation-symmetry axis and the vertical axis, yielding

$$\begin{aligned}\mathcal{H}_{TTI} = \; & d(u_{,x}\cos\theta + u_{,z}\sin\theta)^2 + e(u_{,z}\cos\theta - u_{,x}\sin\theta)^2 \\ & + \; c(u_{,x}\cos\theta + u_{,z}\sin\theta)^2(u_{,z}\cos\theta - u_{,x}\sin\theta)^2 - \; 1.\end{aligned} \tag{5}$$

Similarly, the 3D orthorhombic (OR) Hamiltonian writes under the acoustic approximation

$$\begin{aligned}\mathcal{H}_{OR} =& a(u_{,x})^2 + b(u_{,y})^2 + c(u_{,z})^2 + d(u_{,x})^2(u_{,y})^2 + e(u_{,x})^2(u_{,z})^2 \\ & + f(u_{,y})^2(u_{,z})^2 + g(u_{,x})^2(u_{,y})^2(u_{,z})^2 - 1,\end{aligned} \tag{6}$$

with

$$\begin{cases} a = V_P^2(1+2\epsilon_2), \\ b = V_P^2(1+2\epsilon_1), \\ c = V_P^2, \\ d = V_P^4(1+2\epsilon_2)\big((1+2\epsilon_2)(1+2\delta) - (1+2\epsilon_1)\big), \\ e = -2(\epsilon_2 - \delta_2)V_P^4, \\ f = -2(\epsilon_1 - \delta_1)V_P^4, \\ g = -V_P^6\big((1+2\epsilon_2)^2(1+2\delta) - 2(1+2\epsilon_2)\sqrt{(1+2\delta_2)}\sqrt{(1+2\delta_1)}\sqrt{(1+2\delta)} \\ \qquad + (1+2\delta_2)(1+2\delta_1) - 4(\epsilon_2 - \delta_2)(\epsilon_1 - \delta_1)\big), \end{cases} \tag{7}$$

where anisotropic parameters ϵ_2 and δ_2 hold in the $[x, z]$ plane, anisotropic parameters ϵ_1 and δ_1 hold in the $[y, z]$ plane, and anisotropic parameter δ holds in the $[x, y]$ plane. This parameterization comes from Tsvankin (1997) and is consistent with an alternative parameterization proposed in Alkhalifah (2003) and used in Waheed et al. (2015). The tilted (TOR) case is retrieved by applying the local 3×3 rotation operator involving three rotation angles: the dip angle $\theta(\mathbf{x})$, the azimuth angle $\phi(\mathbf{x})$, and the rotation angle $\psi(\mathbf{x})$ which is the rotation angle of the elastic tensor in the rotated horizontal plane, corresponding to the

crack orientation in this plane. The rotation operator $\mathcal{R}_{\mathrm{TOR}}$ writes

$$\mathcal{R}_{\mathrm{TOR}} = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{8}$$

Replacing quantities $u_{,x}$, $u_{,y}$, and $u_{,z}$ by their rotated expression, obtained from equation 8, into the Hamiltonian of equation 6 yields the TOR Hamiltonian. The substitution is as simple as for going from expression 3 to expression 5, although we do not explicitly write the TOR Hamiltonian here for the sake of concision.

In order to increase the accuracy of the solver, we implement the point-source factorization, as proposed in Pica (1997), Zhang et al. (2005a), Fomel et al. (2009), and Luo and Qian (2011). Le Bouteiller et al. (2018) extend the factorization principles to the DG discretization and show that this makes possible to retrieve a second-order convergence when using approximations by polynomials of order 1. The additive factorization embeds the source singularity inside a reference solution $u_0(\mathbf{x})$ such that

$$u(\mathbf{x}, \xi) = u_0(\mathbf{x}) + \tau(\mathbf{x}, \xi). \tag{9}$$

We then plug expression 9 into Hamiltonian expressions 2, 3, and 6. We adopt the additive factorization since it yields a Hamiltonian which depends only on the gradient of the unknown τ. On the contrary, the Hamiltonian obtained with a multiplicative strategy defined by $u(\mathbf{x}, \xi) = u_0(\mathbf{x})\tau(\mathbf{x}, \xi)$ depends also on the unknown τ itself. The resulting equation would thus contain an additional term leading out of the frame of equation 1.

Finally, the Hamiltonian obtained with the additive factorization is plugged into the dynamic Hamilton–Jacobi equation 1 that we solve for $\tau(\mathbf{x}, \xi)$. This leads to the factored

Hamilton–Jacobi equation for the isotropic case

$$\partial_\xi \tau + \|\nabla_{\mathbf{x}} u_0 + \nabla_{\mathbf{x}} \tau\| - \frac{1}{c} = 0. \tag{10}$$

Factored equations for 2D-VTI and 3D-OR cases are given in Appendix A (equations A-1 and A-2, respectively), and tilted cases are retrieved as described before. The reference solutions u_0 are chosen to be analytical solution in isotropic, TTI, or TOR media with homogeneous elastic parameters given by their values at the source point.

DG discretization

The DG spatial discretization of equation 1 is proposed in Cheng and Wang (2014) and further adapted to the factored TTI Eikonal in Le Bouteiller et al. (2018). We present the scheme in its most general formulation which handles unstructured polygonal (2D) and polyhedral (3D) meshes. Note that the rectangular/cuboid Cartesian formulation simplifies the discretization of the medium following a natural ordering along the x-, y-, and z-axes which will be useful for FSM algorithm as we shall see. The space Ω is partitioned into n elements denoted by $K_i, i = 1, ..., n$. For each element K_i, we choose a local approximation space $\mathcal{P}_i$ spanned by a basis of shape functions $\phi_i^j(\mathbf{x})$. In practice, we consider polynomial spaces P^k containing all polynomials of degree at most k. In our numerical tests, we used P^1 and P^2 spaces. We define $\mathbf{n}_{K_i}$ to be the outward unit normal to the boundary of the element K_i. At element interfaces, traces $v_h^\pm$ and jumps $[v_h]$, of any numerical quantity v_h defined inside two neighboring elements are given respectively by

$$\begin{aligned} v_h^\pm(\mathbf{x}) &= \lim_{\epsilon \downarrow 0} v_h(\mathbf{x} \pm \epsilon \mathbf{n}_{K_i}), \\ [v_h](\mathbf{x}) &= v_h^+(\mathbf{x}) - v_h^-(\mathbf{x}). \end{aligned} \tag{11}$$

With these expressions, as well as their spatial derivatives and their projections on boundaries, different key quantities $\mathcal{F}, \mathcal{G}, \mathcal{K}$ are defined at the boundary between two elements in order to build a causality consistent flux estimation, which is an essential ingredient of the DG formulation. These quantities depend nonlinearly on the solution itself and on the local values of the Hamiltonian function. For the sake of concision, the expressions $\mathcal{F}, \mathcal{G}, \mathcal{K}$ are given in Appendix B.

The weak formulation of equation 1 can be stated as follows:

Find $u_h(.,\xi) \in \{v : v|_{K_i} \in \mathcal{P}_i, \forall i \in \{1,...,n\}\}$ $\forall \xi \geqslant 0$ such that

$$\begin{aligned}
&\int_{K_i} \Big(\partial_\xi u_h(\mathbf{x},\xi) + \mathcal{H}\big(\mathbf{x}, \nabla_\mathbf{x} u_h(\mathbf{x},\xi)\big)\Big) v_i(\mathbf{x}) \mathrm{d}\mathbf{x} \\
&+ \int_{\partial K_i} \mathcal{F}\big(\mathbf{x}, \nabla_\mathbf{x} u_h^{\pm}(\mathbf{x})\big)[u_h](\mathbf{x},\xi) v_i^-(\mathbf{x}) \mathrm{ds} \\
- C\Delta K_i \sum_{S_i^j \in \partial K_i} \frac{1}{\Delta S_i^j} &\int_{S_i^j} \mathcal{G}\big(\mathbf{x}, \nabla_\mathbf{x} u_h^{\pm}(\mathbf{x})\big)[\nabla_\mathbf{x} u_h \cdot \mathbf{n}_{K_i}](\mathbf{x},\xi) v_i^-(\mathbf{x}) \mathrm{ds} \\
- 2C\Delta K_i \sum_{\bar{S}_i^j \in \bar{\partial} K_i} \frac{1}{\Delta \bar{S}_i^j} &\int_{\bar{S}_i^j} \mathcal{K}\big(\mathbf{x}, \nabla_\mathbf{x} u_h^{\pm}(\mathbf{x})\big)\ (\nabla_\mathbf{x} u_h^-(\mathbf{x},\xi) \cdot \mathbf{n}_{K_i})\ v_i^-(\mathbf{x})\ \mathrm{ds} = \ 0,
\end{aligned} \tag{12}$$

for each $i \in \{1,...,n\}$ and for any test function $v_i \in \mathcal{P}_i$,

where ΔK_i (respectively ΔS_i^j) is the volume of the element K_i (respectively the surface of the face j of element K_i). The set ∂K_i denotes the internal faces of element K_i which are shared with other elements. The set $\bar{\partial} K_i$ denotes the external faces of element K_i which are part of the domain boundary $\partial\Omega$. The test functions v_i are shape functions as usual for Galerkin approaches (Zienkewicz and Morgan, 1983). The first term of scheme 12 ensures consistency, embedding a weak formulation of the Hamilton–Jacobi equation. The second term determines the information flow direction and acts as an upwind flux term. It also allows to capture potential shocks, keeping the smallest traveltime next to triplications of the wavefield. The third term treats the so-called rarefaction situations where non-causal

entropy violations may occur in the traveltime solution. The scaling factor C is chosen empirically: numerous observations have shown that 0.25 gives stable schemes in practice (Cheng and Wang, 2014). The fourth term, acting on external edges only, is added in order to enforce suitable radiative boundary conditions (Le Bouteiller et al., 2018). For further technical knowledge about setting up the RK-DG implementation, we refer the reader to Cockburn and Shu (1998).

Local solver

Instead of solving scheme 12 over the whole domain in a time-marching approach as proposed in Le Bouteiller et al. (2018), in this study we evolve elements one by one to a temporary local stationary solution with a local iterative strategy. In other words, given current solutions in its neighbors, solution inside an element evolves until its local steady state is reached, before considering the next element. Therefore, the local solver consists in solving scheme 12 repeatedly for a given element K_i. Formally, we introduce the variable ξ_i as a local pseudo-time variable, which is no longer global. The local integration, which can be considered as a local fixed-point procedure, is performed in ξ_i with an explicit Euler method. Using an explicit time integration is advantageous because scheme 12 is highly nonlinear: the numerical fluxes depend on the solution itself in a nonlinear way. Moreover, we verify that it is not necessary to use higher-order time schemes like second-order Runge–Kutta method in this local solver. Its implementation is straightforwardly derived from the RK-DG formulation, taking care of restricting the computation to the current element. Iterations are performed until a convergence criterion is reached, which we refer to as the real quantity *local_conv*.

However, in some transient configurations, such convergence may not be reached due to the nonlinearity of the problem and causality considerations. When the upwind flux is initially oriented toward a given direction at an edge of a cell (e.g. an initial orientation given by a direct wave), the cell could be in a configuration such that changing its orientation to another direction (e.g. a diving wave coming through a higher velocity zone) turns out to be impossible during the fixed-point local procedure. Once the local solver has reached a given number *max_local_iter* of local iterations without local convergence, we have designed a specific procedure, that we call *degenerate local solver*, which is activated for overcoming such situation. This degenerate solver detects, among the four (2D) or six (3D) neighbors of the current element, the one with the lowest traveltime values (in practice we compare the maximum values at edges). Once the fastest edge has been retrieved, we simulate a (non-physical) plane wave coming from this edge and traveling through the element at the local wave speed. This allows to reconstruct a temporary solution with a correct orientation of fluxes and respecting the velocity inside the element. Such a solution will not be the final solution after the sweeping strategy we describe in the next paragraph. This temporary simple estimation enables various branches of the final solution to propagate along the current sweep and eventually to keep the fastest one at each location. We proceed with sweeps until this degenerate solver is not activated anymore: this is one of the criteria required for global convergence.

The degenerate local solver we have designed is a key procedure which unlocks several crucial issues for developing an efficient FSM-DG method. First, it avoids to be trapped in a wrong causality setting, as described above. Second, as a consequence, it exempts from the need of a good initial solution, which was a practical limitation in Zhang et al. (2011). Finally, it exempts from a severe constraint on the Hamiltonian: the RK-DG scheme used in

Le Bouteiller et al. (2018) required a Lipschitz continuous Hamiltonian in order to define a suitable CFL condition which would not depend on the solution. This limitation prompted the authors to use a complicated 2D TTI Hamiltonian in order to ensure stability in all the elements at every timestep. In our FSM-DG method, if an instability raises during the local integration, then the degenerate solver acts as a post-treatment limiter, by replacing the solution with an approximate one respecting the local causality. Therefore, we are able to use standard anisotropic Hamiltonians of equations 3 and 6.

Fast-sweeping algorithm

The global solution is obtained by applying the local solver successively to all the elements in a block Gauss–Seidel approach: nonlinear equations are solved element by element, each element representing several unknowns (degrees of freedom). As in Zhao (2005), the alternating sweepings follow the four (2D) or eight (3D) natural orderings of the structured Cartesian mesh. For unstructured grids, it could be possible to pre-compute specific orderings of the elements for sweeping: we have not implemented such strategy because of the additional implementation complexity (Qian et al., 2007a,b). For Cartesian grid, the natural directions allow to sample efficiently the characteristics of the Eikonal equation.

A boundary condition is set at the source: inside the source element, the numerical solution τ is set to zero and does not evolve, so that the traveltime solution equals the reference solution inside this source element.

Initialization

Initialization steps are defined for both local solver and sweeping procedure. Regarding the sweeping, the first four (2D) or eight (3D) sweeps are performed from the source element toward the boundaries, respectively in the four (2D) or eight (3D) quarters of the domain defined by the horizontal and vertical axes aligned with the source point. We have found this to be the best initialization in order to optimize the number of sweeps needed to reach the convergence. At the local scale, the first time the local solver is called for a given element, the degenerate solver is executed at first, then the local iterative procedure occurs. The element is tagged as updated afterwards. When executing the local solver, only the at-least-once updated neighbors are considered. At edges where a neighbor has not been updated yet, a boundary condition is applied, which is the same as at the domain boundaries (fourth term in scheme 12).

For sake of conciseness, the algorithms are presented in 2D. The block Gauss–Seidel procedure and its initialization are detailed in Algorithms 1 and 2. The structure of the local solver is presented in Algorithm 3, and the degenerate solver is detailed in Algorithm 4. The extension to higher dimensions of these algorithms is straightforward by sweeping in the additional dimensions.

Mesh deformation for topography

Finite-element methods are able to handle complex geometries in a natural way, thanks to the integral formulation allowing deformed elements. Scheme 12 is written in a general formulation where cells could be of any polygonal type. For example, we could design 2-D unstructured triangle meshes, as shown in Le Bouteiller et al. (2018), Case Study 4.

Instead, at the exploration scale, vertically deformed Cartesian grids are very attractive for their simplicity (Hestholm, 1999; Tarrass et al., 2011; Trinh et al., 2018). The main advantages are that there is no mesh generation step to be performed by the user: the spatial position of each element can be accessed directly by its indices in x, y, and z. Moreover, the grid allows to keep the natural ordering of elements when performing the FSM algorithm, yielding an optimal efficiency for the Gauss-Seidel procedure. The topography variation is simply described by a gradual vertical deformation of elements, keeping a constant number of elements in x, y, and z directions. After interpolating the topography $z^\star$ over the (x, y) nodes, the explicit mapping between a reference unit cube and each deformed element is established. The Gauss points used for computing integrals in scheme 12 are defined in the reference cube, and the quantities in the physical space are estimated at these Gauss points using the mapping, the local Jacobian, and chain rules for the spatial derivatives. The quantities needed at Gauss points are precomputed prior to solving Eikonal itself, and only once for a given deformed grid. Figure 1 shows an example of a vertically deformed mesh in 2D.

NUMERICAL RESULTS

All the numerical experiments (2D and 3D) are performed on a laptop computer using a single core of an Intel® Core™ i7-4600U computer processing unit with a frequency of 2.10 GHz, and 8 GB of DDR3 SDRAM. The computations are done in double precision.

Algorithm 1 Sweeping

```
1: procedure SWEEP1
2:     for i = 1 → N_x do
3:         for j = 1 → N_z do
4:             call LOCAL_SOLVER(i, j)
5: procedure SWEEP2
6:     for i = N_x → 1 do
7:         for j = 1 → N_z do
8:             call LOCAL_SOLVER(i, j)
9: procedure SWEEP3
10:    for i = N_x → 1 do
11:        for j = N_z → 1 do
12:            call LOCAL_SOLVER(i, j)
13: procedure SWEEP4
14:    for i = 1 → N_x do
15:        for j = N_z → 1 do
16:            call LOCAL_SOLVER(i, j)
17: procedure SWEEPING
18:    call INIT_SWEEP
19:    for k = 1 → max_iter do
20:        select case mod(k − 1, 4) + 1
21:            case(1) call SWEEP1
22:            case(2) call SWEEP2
23:            case(3) call SWEEP3
24:            case(4) call SWEEP4
25:        if ‖(u − u_old)/u_old‖ < global_conv then return
```

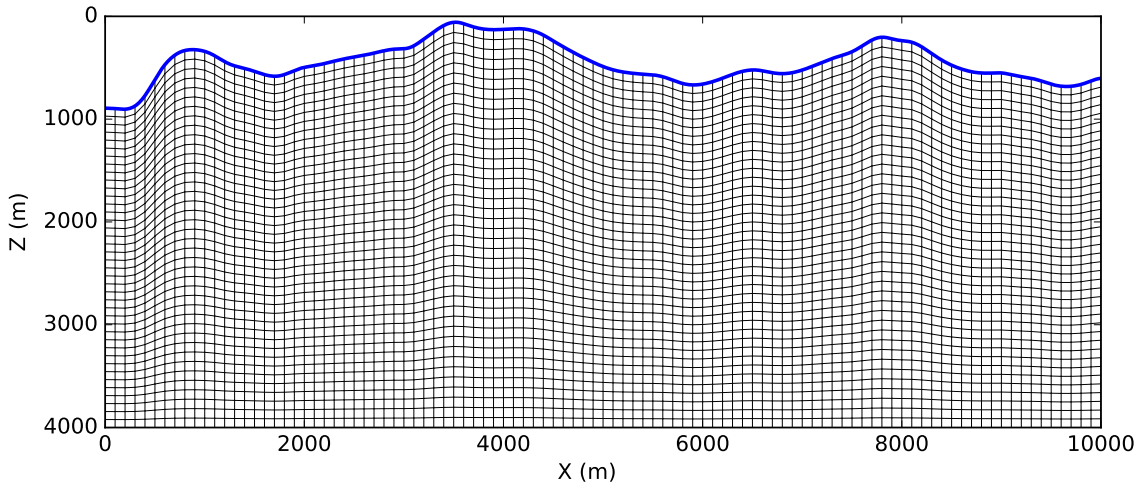

Figure 1: Vertically deformed mesh for a cross-section extracted from SEAM II Foothills benchmark model. The topography (blue line) has been interpolated over 101 equally-distributed points in x-direction. The mesh is 100×40 elements.

Algorithm 2 Init Sweeping

```
1:  procedure INIT_SWEEP
2:      updated(:,:)=False
3:      for i = ix_s → N_x do
4:          for j = iz_s → N_z do
5:              call LOCAL_SOLVER(i, j)
6:              updated(i, j)=True
7:      for i = ix_s → 1 do
8:          for j = iz_s → N_z do
9:              call LOCAL_SOLVER(i, j)
10:             updated(i, j)=True
11:     for i = ix_s → 1 do
12:         for j = iz_s → 1 do
13:             call LOCAL_SOLVER(i, j)
14:             updated(i, j)=True
15:     for i = 1 → ix_s do
16:         for j = iz_s → 1 do
17:             call LOCAL_SOLVER(i, j)
18:             updated(i, j)=True
```

Algorithm 3 Local Solver

procedure LOCAL_SOLVER(i, j)
  for $n = 1 \rightarrow max\_local\_iter$ **do**
    $inside$ = HJ_integral(i, j)
    for $(k, l) \in$ neighbors(i, j) **do**
      if $updated(k, l)$ **then**
        $edges \leftarrow edges +$ flux_integral(i, j, k, l)
      else
        $edges \leftarrow edges +$ boundary_integral(i, j, k, l)
    $u^{ij} = u^{ij}_{old} + \Delta t(inside + edges)$
    if $\|(u^{ij} - u^{ij}_{old})/u^{ij}_{old}\| < local\_conv$ **then return**
    $u^{ij}_{old} = u^{ij}$
  call DEGENERATE_SOLVER(i, j)

Algorithm 4 Degenerate Solver

procedure DEGENERATE_SOLVER(i, j)
  $t_{\min} = 100000$
  for $(k, l) \in$ neighbors(i, j) **do** $t_{\min} = \min(t_{\min}, \max(u^{kl}_{\text{edge}}))$
  $u^{ij} =$ DG_projection$\Big(t_{\min} + \text{dist}(\mathbf{x}, \text{edge_min})/\min_{K_{ij}}(c(\mathbf{x}))\Big)$

Vertical gradient of velocity

We first exhibit the efficiency of the FSM-DG technique in a 2D isotropic case, where the velocity of the medium varies linearly with depth. In a 4 × 4 km square, the velocity varies from a value of 2 km/s at the surface to a value of 4 km/s at depth. The point source is located at the surface with coordinates $x_s = 2$ km, $z_s = 0$ km. The knowledge of the exact solution enables error computation for traveltime as well as its spatial derivatives (Fomel et al., 2009). A refinement study is carried out and the results are shown in Tables 1 and 2: L^2 errors are exhibited with respect to the number of degrees of freedom when using P^1 polynomial approximations, together with a comparison of CPU times between the RK-DG and the FSM-DG techniques. We perform similar experiments using a fast-sweeping FD

RK-DG						
N	#dof	Error	Order	dx error	Order	CPU (s)
21	1323	$1.74\text{E}-03$		$1.13\text{E}-02$		0.36
41	5043	$4.66\text{E}-04$	1.97	$5.51\text{E}-03$	1.08	1.6
81	19683	$1.21\text{E}-04$	1.98	$2.71\text{E}-03$	1.04	11
161	77763	$3.07\text{E}-05$	1.99	$1.34\text{E}-03$	1.02	78
321	309123	$7.74\text{E}-06$	2.00	$6.69\text{E}-04$	1.01	600
641	1232643	$1.94\text{E}-06$	2.00	$3.34\text{E}-04$	1.00	4900

Table 1: RK-DG results from the first numerical example in 2D. Number of elements along one direction (N), number of degrees of freedom (#dof), L^2 error of the solution and its derivative along x-direction, convergence orders, and CPU times, for P^1 polynomial approximation.

solver from Noble et al. (2014); results are given in Table 3. All these results are compared in Figure 2.

The convergence of the schemes are highlighted in Figure 2a. The FD method exhibits a first-order convergence: when the spatial discretization step is divided by two, the error is also divided by two. Note that both RK-DG and FSM-DG methods yield the same error, since they yield the same final state in a given discretization after integration in ξ. As expected, the higher slope of decrease of the error with respect to the number of degrees of freedom highlights the second-order convergence of the P^1 DG approximation. Coherently, the x-derivative of the solution exhibits a first-order convergence (Tables 1 and 2). This result was already exhibited in Le Bouteiller et al. (2018). In terms of computational cost,

FSM-DG							
N	#dof	Error	Order	dx error	Order	CPU (s)	CPU ratio
21	1323	$1.73\text{E}-03$		$1.18\text{E}-02$		0.21	1.7
41	5043	$4.65\text{E}-04$	1.96	$5.60\text{E}-03$	1.11	0.43	3.7
81	19683	$1.21\text{E}-04$	1.98	$2.73\text{E}-03$	1.06	0.68	16
161	77763	$3.07\text{E}-05$	1.99	$1.35\text{E}-03$	1.03	2.0	39
321	309123	$7.74\text{E}-06$	2.00	$6.70\text{E}-04$	1.01	6.6	91
641	1232643	$1.95\text{E}-06$	1.99	$3.35\text{E}-04$	1.00	26	188

Table 2: FSM-DG results from the first numerical example in 2D. Number of elements along one direction (N), number of degrees of freedom (#dof), L^2 error of the solution and its derivative along x-direction, convergence orders, CPU times, and CPU ratio between RK-DG and FSM-DG, for P^1 polynomial approximation. Please note the slow increase of the CPU time with respect to the number of degrees of freedom (see Figure 2b).

FSM-FD				
N	#dof	Error	Order	CPU (s)
72	5184	$6.08\text{E}-03$		0.01
140	19600	$2.91\text{E}-03$	1.06	0.05
278	77284	$1.46\text{E}-03$	0.99	0.18
556	309136	$7.32\text{E}-04$	1.00	0.76
1110	1232100	$3.67\text{E}-04$	1.00	3.1
2220	4928400	$1.84\text{E}-04$	1.00	12
4440	19713600	$9.42\text{E}-05$	0.97	50

Table 3: FSM-FD results from the first numerical example in 2D. Number of elements along one direction (N), number of degrees of freedom (#dof), L^2 error of the solution, convergence orders, and CPU times.

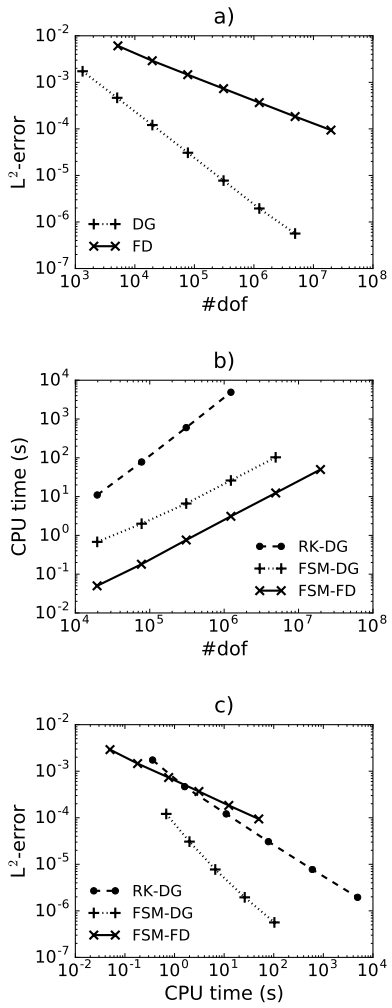

Figure 2: Comparison of FD and DG methods with respect to accuracy (L^2 error), CPU time and number of degrees of freedom. a) The DG scheme enables high-order approximation, yielding lower error for a fixed number of degrees of freedom compared to the FD method. b) Although the DG method implies a higher computational burden for a fixed number of degrees of freedom compared to the FD method, the fast-sweeping algorithm applied to the DG method (FSM-DG) exhibits a linear complexity, which is more efficient than the RK-DG method. c) Finally, the resulting efficiency is higher with the FSM-DG method than the FD method: reaching a fixed level of error is done with lower CPU time.

the analysis of the slopes in Figure 2b shows that the RK-DG method has a computational complexity in $\mathcal{O}(\#\text{dof}^{3/2})$, compared to the linear complexity ($\mathcal{O}(\#\text{dof})$) of the FSM-FD method. This is why the RK-DG method is not efficient. However, the new FSM-DG algorithm constitutes a huge improvement since it exhibits a linear complexity in $\mathcal{O}(\#\text{dof})$, as the FSM-FD scheme. Finally, the analysis of Figure 2c underlines that, for reaching low levels of error, the most efficient algorithm is the new FSM-DG. For a given level of error, this algorithm is much faster than FSM-FD because it needs less degrees of freedom to reach the same accuracy.

In terms of memory requirements, in this 2D case, the FSM-FD code requires approximately 315 MB of memory for the case where $N = 4440$. This amount is obtained by summing the sizes of the arrays allocated for the traveltime and for the velocity model. It corresponds to 16 bytes per degree of freedom. In our current implementation, the DG codes require approximately 115 MB of memory for the case $N = 641$. Since there are three degrees of freedom per element, it corresponds to 90 bytes per degree of freedom. The difference with the FD code comes from additional arrays allocated in memory for the values of the reference solution u_0 and its spatial derivatives at all the Gauss points. This is designed as such in order to optimize the CPU time. Alternatively, these quantities could be computed on the fly. Note that the FSM-DG and RK-DG codes have the same memory requirements.

Keeping the vertical gradient of velocity, we now perform similar simulations on a 3D $4 \times 4 \times 4$ km cube with the FSM-DG method. The point source is located at the surface with coordinates $x_s = 2$ km, $y_s = 2$ km, $z_s = 0$ km. The results obtained with a P^1 approximation are detailed in Table 4 and highlighted in Figure 3. As expected, we retrieve a second-order convergence of the 3D DG discretization (Figure 3a). Moreover, the FSM-

FSM-DG						
N	#dof	Error	Order	dx error	Order	CPU (s)
11	5324	$1.41\text{E}-03$		$2.06\text{E}-02$		2.4
21	37044	$3.73\text{E}-04$	2.05	$1.11\text{E}-02$	0.96	4.3
41	275684	$9.59\text{E}-05$	2.03	$5.76\text{E}-03$	0.98	15
81	2125764	$2.43\text{E}-05$	2.01	$2.93\text{E}-03$	0.99	133
161	16693124	$6.07\text{E}-06$	2.02	$1.48\text{E}-03$	0.99	1150

Table 4: FSM-DG results from the first numerical example in 3D. Number of elements along one direction (N), number of degrees of freedom (#dof), L^2 error of the solution and its derivative along x-direction, convergence orders, and CPU times, for P^1 polynomial approximation.

DG method still exhibits a linear complexity (Figure 3b), which makes it very efficient in 3D as well.

In terms of memory requirements, in the 3D case with $N = 161$, the FSM-DG codes require approximately 3500 MB of memory, yielding 200 bytes per degree of freedom. Here again, the values of the reference solution u_0 and its spatial derivatives at all the Gauss points are stored in memory.

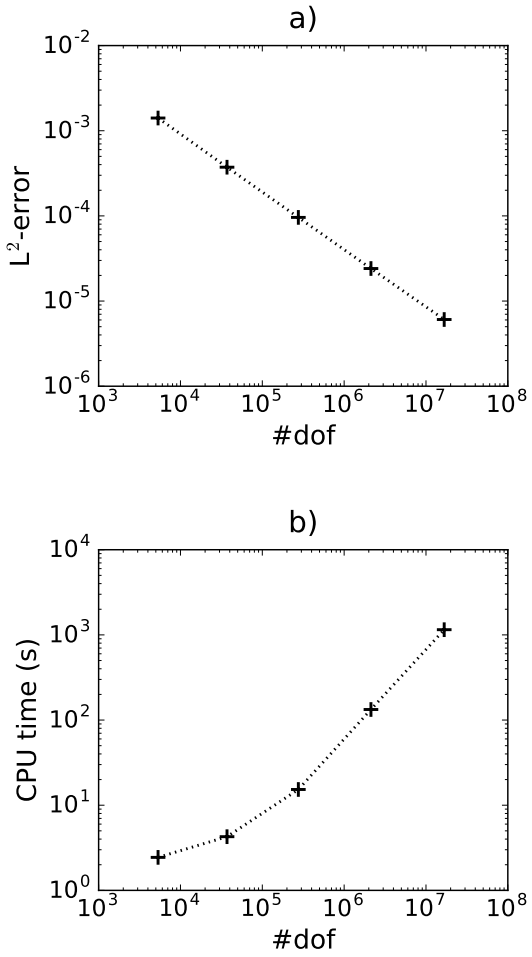

Figure 3: FSM-DG results from the first numerical example in 3D: L^2 error and CPU time with respect to the number of degrees of freedom. The curved shape for small number of degrees of freedom when the CPU time is lower than 10 seconds is explained by initialization steps in the code that do not depend on the size of the mesh, such as model reading and source handling, for instance.

Homogeneous 3D tilted orthorhombic medium with complex topography

In this example, we consider a homogeneous 3D TOR medium, with anisotropic parameters chosen as follows:

$$\begin{cases} V_P = 2000\,\mathrm{m/s}, \\ \epsilon_2 = 0.2, \delta_2 = 0.1, \\ \epsilon_1 = 0.4, \delta_1 = 0.3, \\ \delta = 0.1, \\ \phi = 30^\circ, \\ \theta = 45^\circ, \\ \psi = -15^\circ. \end{cases} \tag{13}$$

We define a physical domain with a complex topography using a part of the SEAM II model (Regone et al., 2017). We consider a domain of size 3 km along the x-axis, 7 km along the y-axis, and 2 km along the z-axis. This domain is restricted in the z-direction by a topography provided with the model, which we amplify by a factor 2 in order to clearly exhibit its imprint on the traveltimes. We build a vertically-deformed mesh from a regular Cartesian grid to follow this topography. The mesh spacing before deformation is 100 m in the three directions. Traveltimes are computed both with and without the topography, for a point-source located at $x = 1150$ m, $y = 50$ m, $z = 945$ m. Both results are superimposed in Figure 4. The imprint of the topography is clearly visible in the near-surface areas and might be understood by applying the Huygens principle at the bottom points of the topography, from which the upper parts of the domain are illuminated. This example shows the importance of taking care of topography, and the good behavior of the FSM-DG method when doing so. Let us emphasize that not only the traveltimes are modified by the

topography, but also the spatial derivatives, which indicate the front propagation direction. These derivatives are of major interest when considering subsequent amplitude or angle estimations.

3D complex SEAM II model

In this example, our FSM-DG scheme is applied onto the same part of the isotropic SEAM II model (Regone et al., 2017). Here, we consider the original topography as provided with the model (no amplification factor). To do so, we build a vertically-deformed mesh as in the previous example: The original mesh step is 100 m in the three directions, and the resulting deformed mesh is shown in Figure 5. The P-wave velocity model, shown in Figure 6, exhibits complex velocity structures with layering and faulting, as well as near-surface low-velocity areas which are known to impair imaging and inversion results if not carefully considered. We use the P-wave velocity of the model to compute first-arrival traveltimes for a source located at $x = 375$ m, $y = 350$ m, and $z = 528$ m. The traveltime isocontours are superimposed over the P-wave velocity model in Figure 6. In order to illustrate the quality of the result, we perform a computation using a spectral-element solver of the elastic wave equation applied to the same model. For this computation, we use the code SEM46 (Trinh et al., 2017) with an impulse source in a $60 \times 140 \times 40$ mesh. In Figure 7 and 8, the traveltime isocontours obtained by the FSM-DG method are superimposed over the displacement field in the x-direction obtained with the SEM46 code, in vertical and horizontal planes, highlighting an almost perfect agreement between the first-arrival traveltimes and the wavefront, although they are based on different equations (Eikonal equation versus elastic wave equation) and different numerical methods (Discontinuous Galerkin method versus spectral element method), respectively.

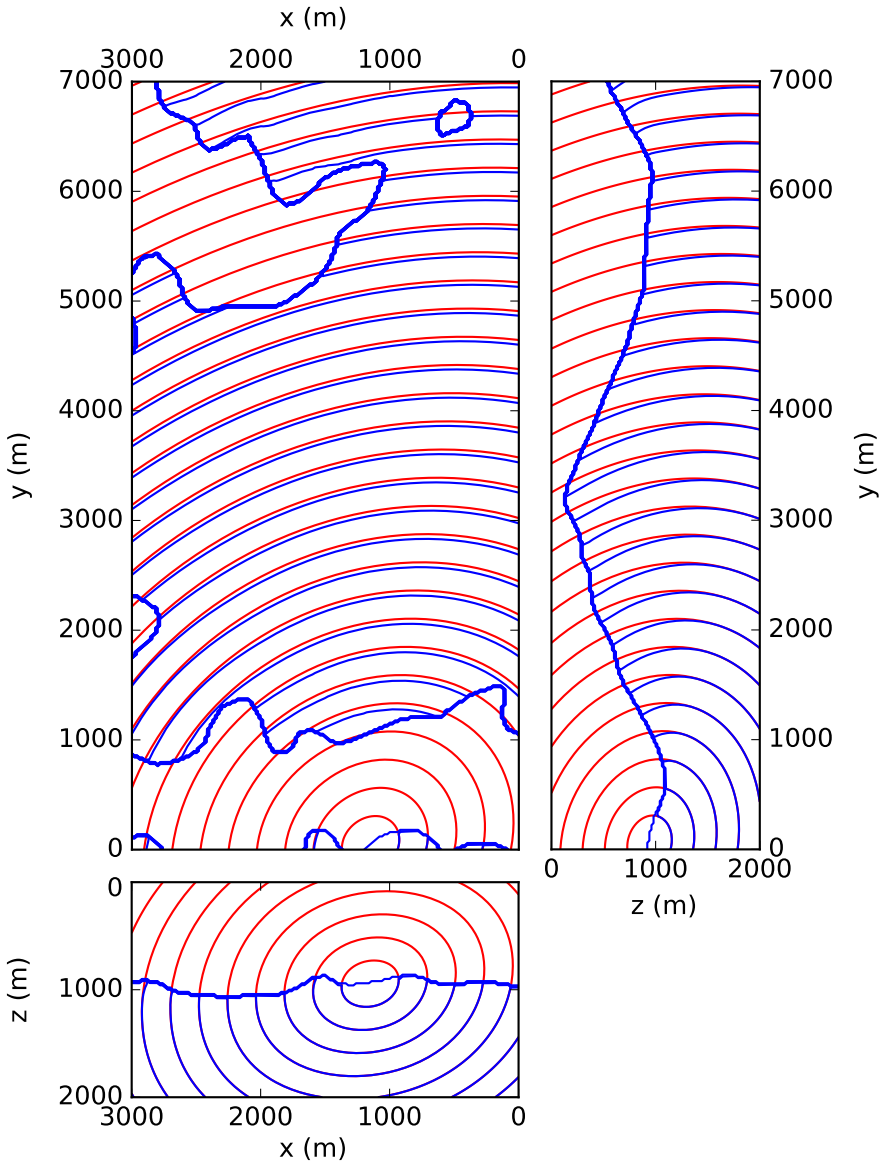

Figure 4: Traveltime isocontours computed in the homogeneous TOR model with (blue) and without (red) the topography. Panels at $z = 945$ m (left), $x = 1150$ m (right), and $y = 50$ m (bottom). Please note that the thick zone is the contour of the topography: the blue solution is not built above the topography, while the red solution assumes a flat topography at the top of the domain.

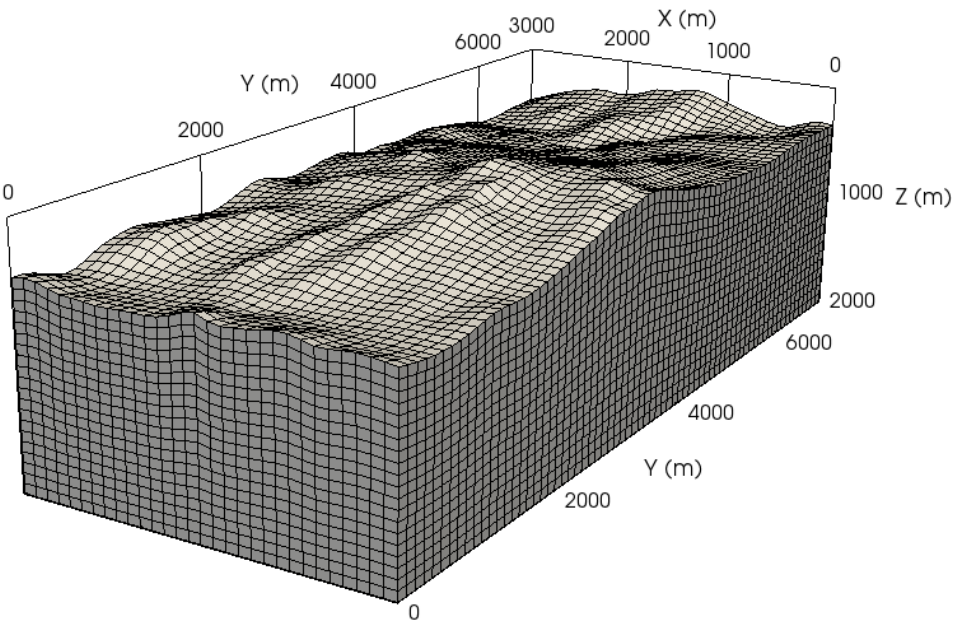

Figure 5: The 30x70x20 vertically-deformed mesh built for the SEAM II model.

In presence of a non-flat topography, the memory requirements are higher than in the Cartesian 3D case. The last example requires approximately 180 MB of memory, corresponding to 1000 bytes per degree of freedom. This is mainly due to the values of the derivatives of the basis functions that we keep in memory at each Gauss point. Alternatively, these values could be computed on-the-fly.

CONCLUSION

A FSM algorithm has been introduced in a DG approach for solving the 3D Eikonal equation. The DG approach had been previously introduced in a 2D framework, using a Runge–Kutta solver, responsible for a high computational complexity in $\mathcal{O}(\#\text{dof}^{3/2})$. The new

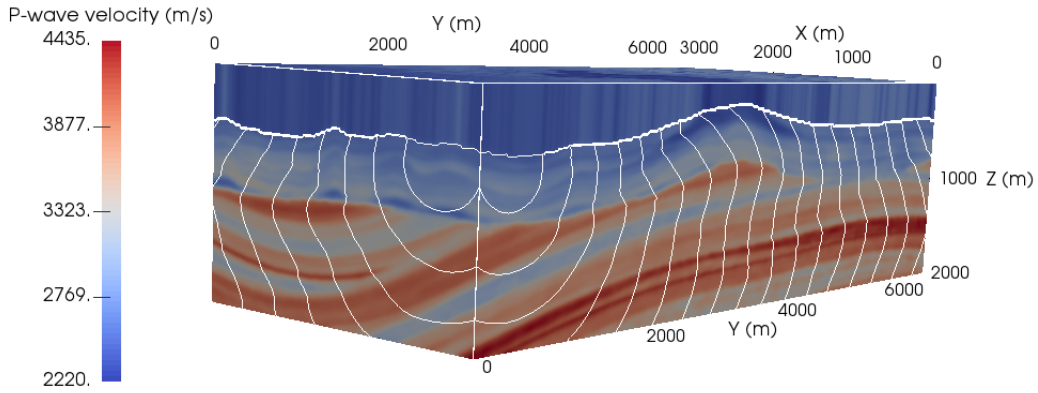

Figure 6: The Vp model extracted from the SEAM II model, with traveltime isocontours superimposed.

FSM algorithm provides a significant increase of efficiency, making possible to reach a linear complexity as for FSM-FD approach, while benefiting for the high accuracy and higher-order convergence rate associated with DG approach. For this reason, higher accuracy for the traveltime solution and its spatial derivatives is obtained compared to FD methods, while complex structures are handled in a stable and accurate way, thanks to FE properties. This is illustrated by the use of deformed Cartesian grid for handling topography. Both 2D and 3D implementations are performed, with TTI and TOR anisotropy, thanks to the general Hamiltonian formulation of the DG scheme. Even more general Hamiltonian could be considered in the future, accounting for instance for triclinic anisotropy.

We may now use these results together with accurate solvers for seismic amplitudes and take-off angles computations in order to revisit asymptotic inversion (traveltime/slope tomography) and imaging approaches (quantitative migration using amplitudes and angles).

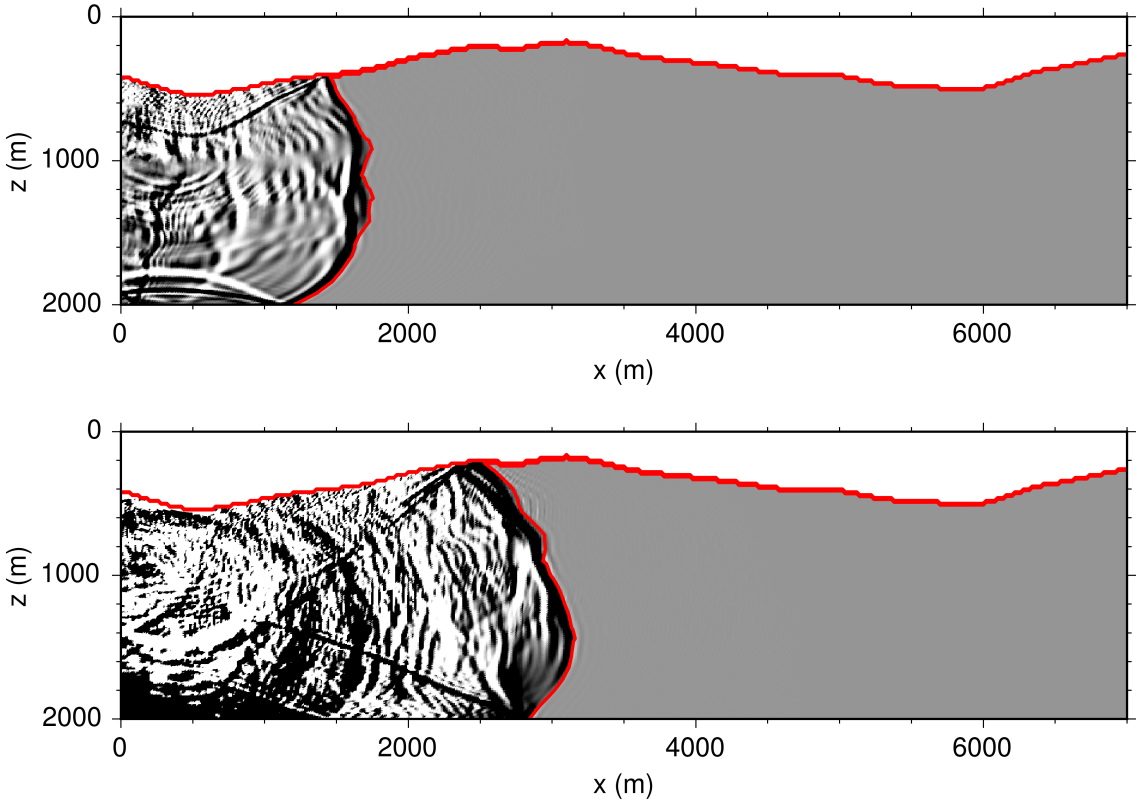

Figure 7: Displacement field in the x-direction from SEM46, an elastic spectral-element solver, with an impulse source. The traveltime isocontours are superimposed. Snapshots in the plane $x = 1500$ m and corresponding isocontours at time 0.6 (top) and 0.9 (bottom) seconds. One could notice the numerical noise occurring ahead of the front, which comes from the wave propagation solution, and which is visible due to the saturation of the plot.

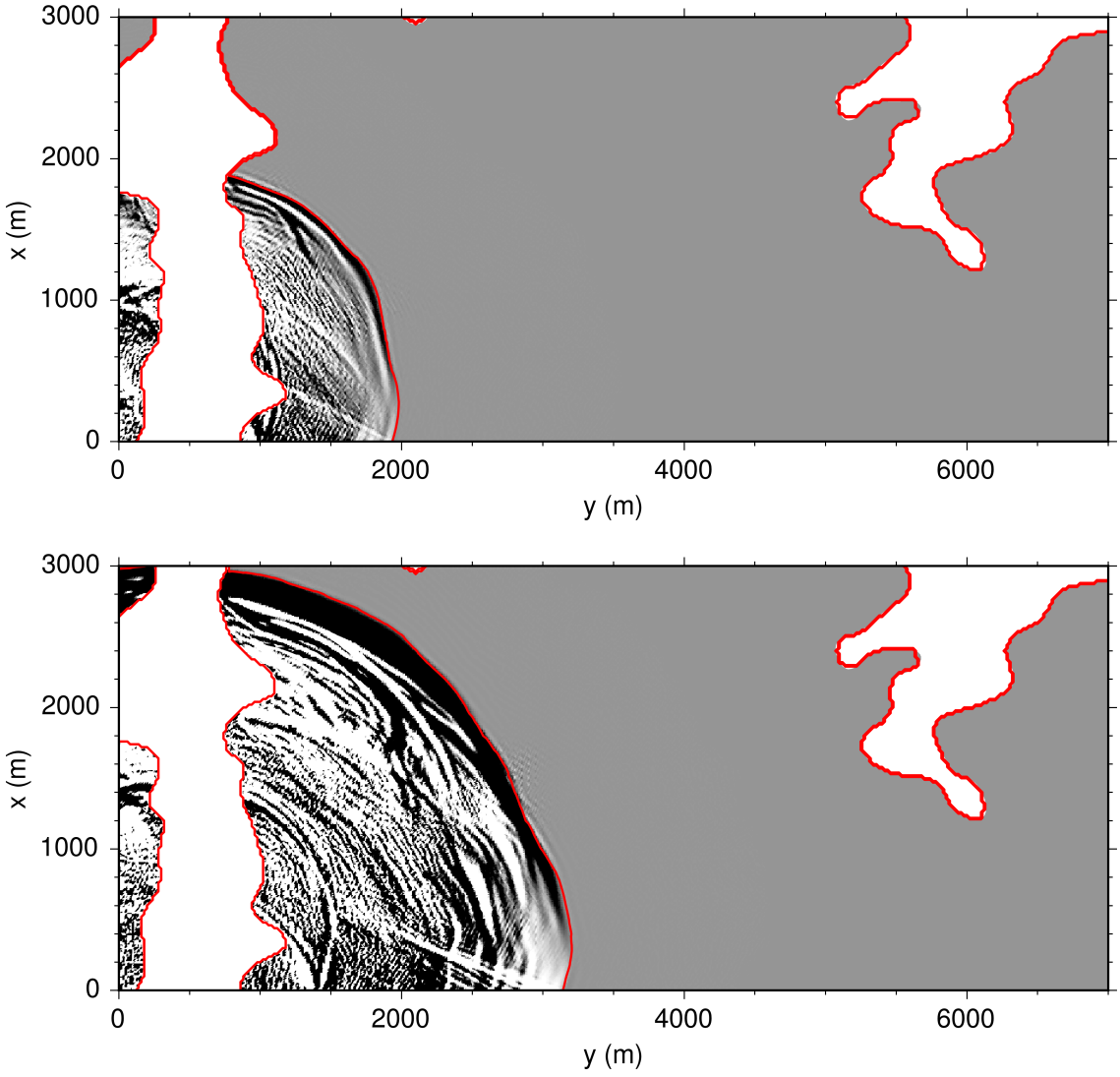

Figure 8: Displacement field in the x-direction from SEM46, an elastic spectral-element solver, with an impulse source. The traveltime isocontours are superimposed. Snapshots in the plane $z = 500$ m and corresponding isocontours at time 0.6 (top) and 0.9 (bottom) second. The topographical contour is expressed by the white zone where no solution is computed. Here again, one could notice the numerical noise occurring ahead of the front.

ACKNOWLEDGMENTS

This study was partially funded by the SEISCOPE consortium (*http://seiscope2.osug.fr*), sponsored by AKERBP, CGG, CHEVRON, EXXON-MOBIL, JGI, PETROBRAS, SCHLUMBERGER, SHELL, SINOPEC, STATOIL and TOTAL. This study was granted access to the HPC resources of the Froggy platform of the CIMENT infrastructure (*https://ciment.ujf-grenoble.fr*), which is supported by the Rhône-Alpes region (GRANT CPER07_13 CIRA), the OSUG@2020 labex (reference ANR10 LABX56) and the Equip@Meso project (reference ANR-10-EQPX-29-01) of the programme Investissements d'Avenir supervised by the Agence Nationale pour la Recherche, and the HPC resources of CINES/IDRIS/TGCC under the allocation 046091 made by GENCI.

APPENDIX A: FACTORED HAMILTON–JACOBI EQUATIONS

Plugging expression 9 into 2D-VTI Hamiltonian expression 3, and finally into the dynamic Hamilton–Jacobi equation 1 that we solve for $\tau(\mathbf{x}, \xi)$, leads to the factored 2D-VTI Hamilton–Jacobi equation

$$\partial_\xi \tau + d(u_{0,x} + \tau_{,x})^2 + e(u_{0,z} + \tau_{,z})^2 + c(u_{0,x} + \tau_{,x})^2 d(u_{0,z} + \tau_{,z})^2 - 1 = 0. \tag{A-1}$$

Now, using the 3D orthorhombic Hamiltonian of equation 6, the factored 3D-OR Hamilton–Jacobi equation writes

$$\begin{aligned} &\partial_\xi \tau + a(u_{0,x} + \tau_{,x})^2 + b(u_{0,y} + \tau_{,y})^2 + c(u_{0,z} + \tau_{,z})^2 \\ &+d(u_{0,x} + \tau_{,x})^2 (u_{0,y} + \tau_{,y})^2 + e(u_{0,x} + \tau_{,x})^2 (u_{0,z} + \tau_{,z})^2 + f(u_{0,y} + \tau_{,y})^2 (u_{0,z} + \tau_{,z})^2 \\ &+g(u_{0,x} + \tau_{,x})^2 (u_{0,y} + \tau_{,y})^2 (u_{0,z} + \tau_{,z})^2 - 1 = 0. \end{aligned} \tag{A-2}$$

APPENDIX B: DETAILED DG SCHEME

At an interface of element K_i, we define a two-component vector by the expression

$$\nabla_{\mathbf{x}} u^{\pm}_{h_{K_i}} = \begin{pmatrix} (\nabla_{\mathbf{x}} u_h \cdot \mathbf{n}_{K_i})^{\pm} \\ \\ \overline{\nabla_{\mathbf{x}} u_h \cdot \mathbf{t}_{K_i}} \end{pmatrix}. \tag{B-1}$$

The first component $(\nabla_{\mathbf{x}} u_h \cdot \mathbf{n}_{K_i})^{\pm}$ is the projection onto the normal $\mathbf{n}_{K_i}$, of the gradient of the numerical solution computed inside the K_i cell $(-)$, or inside its neighbor $(+)$. The second component $\overline{\nabla_{\mathbf{x}} u_h \cdot \mathbf{t}_{K_i}}$ holds for the mean of the projections onto the tangential vector $\mathbf{t}_{K_i}$ of the gradient of the numerical solution computed inside the K_i cell and inside its corresponding neighbor.

We also introduce the following quantities:

$$
\begin{aligned}
\mathcal{H}_{K_i}^{\pm} &= \mathcal{H}\big(\mathbf{x}^{\pm}, \nabla_{\mathbf{x}} u_{h_{K_i}}^{\pm}\big), \\
\mathcal{H}_{\mathbf{n}_{K_i}} &= \nabla_{\nabla u}\mathcal{H} \cdot \mathbf{n}_{K_i}, \\
\mathcal{H}_{\mathbf{n}_{K_i}}^{\pm} &= \mathcal{H}_{\mathbf{n}_{K_i}}\big(\mathbf{x}^{\pm}, \nabla_{\mathbf{x}} u_{h_{K_i}}^{\pm}\big), \\
\tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x}) &= \begin{cases} \frac{\mathcal{H}_{K_i}^{+} - \mathcal{H}_{K_i}^{-}}{[\nabla_{\mathbf{x}} u_h \cdot \mathbf{n}_{K_i}](\mathbf{x})}, & \text{if } [\nabla_{\mathbf{x}} u_h \cdot \mathbf{n}_{K_i}](\mathbf{x}) \neq 0, \\ \frac{1}{2}\left(\mathcal{H}_{\mathbf{n}_{K_i}}^{+} + \mathcal{H}_{\mathbf{n}_{K_i}}^{-}\right), & \text{otherwise,} \end{cases} \\
\delta_{\mathbf{n}_{K_i}}(\mathbf{x}) &= \max\big(0, \tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x}) - \mathcal{H}_{\mathbf{n}_{K_i}}^{-}, \mathcal{H}_{\mathbf{n}_{K_i}}^{+} - \tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x})\big), \\
\chi_{\mathbf{n}_{K_i}}(\mathbf{x}) &= \max\big(\delta_{\mathbf{n}_{K_i}}(\mathbf{x}), |\tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x})|\big).
\end{aligned}
\tag{B-2}
$$

Quantities $\mathcal{F}, \mathcal{G}, \mathcal{K}$ introduced in scheme 12 now write

$$
\begin{aligned}
\mathcal{F} &= \min\big(\tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x}), 0\big), \\
\mathcal{G} &= \chi_{\mathbf{n}_{K_i}}(\mathbf{x}) - |\tilde{\mathcal{H}}_{\mathbf{n}_{K_i}}(\mathbf{x})|, \\
\mathcal{K} &= \min\big(\mathcal{H}_{\mathbf{n}_{K_i}}^{-}(\mathbf{x}), 0\big).
\end{aligned}
\tag{B-3}
$$

The key quantity for preserving causality is $\tilde{\mathcal{H}}_{\mathbf{n}_K}$, referred to as the Roe speed: its sign specifies the information flow direction at an interface between two cells. Thanks to the min operator in $\mathcal{F}$, when computing the second integral of scheme 12, the *downwind* cell receives information from the *upwind* cell, while the *upwind* cell is not affected by the *downwind* cell information. The continuity between elements is thus weakly enforced in an upwind manner.

REFERENCES

Alkhalifah, T., 2000, An acoustic wave equation for anisotropic media: Geophysics, **65**, 1239–1250.

———, 2003, An acoustic wave equation for orthorhombic anisotropy: Geophysics, **68**, 1169–1172.

———, 2011, Efficient traveltime compression for 3D prestack Kirchhoff migration: Geophysical Prospecting, **59**, 1–9.

Beylkin, G., 1985, Imaging of discontinuities in the inverse scaterring problem by inversion of a causal generalized Radon transform: Journal of Mathematical Physics, **26**, 99–108.

Beylkin, G., and R. Burridge, 1990, Linearized inverse scattering problems in acoustics and elasticity: Wave motion, **12**, 15–52.

Billette, F., and G. Lambaré, 1998, Velocity macro-model estimation from seismic reflection data by stereotomography: Geophysical Journal International, **135**, 671–680.

Bleistein, N., 1987, On the imaging of reflectors in the Earth: Geophysics, **52**, 931–942.

Boué, M., and P. Dupuis, 1999, Markov chain approximations for deterministic control problems with affine dynamics and quadratic cost in the control: SIAM Journal on Numerical Analysis, **36**, 667–695.

Cao, S., and S. Greenhalgh, 1994, Finite-difference solution of the eikonal equation using an efficient, first-arrival, wavefront tracking scheme: Geophysics, **59**, 632–643.

Červený, V., 2001, Seismic ray theory: Cambridge University Press.

Cheng, Y., and C.-W. Shu, 2007, A discontinuous Galerkin finite element method for directly solving the Hamilton-Jacobi equations: Journal of Computational Physics, **223**, 398 – 415.

Cheng, Y., and Z. Wang, 2014, A new discontinuous Galerkin finite element method for directly solving the Hamilton–Jacobi equations: Journal of Computational Physics, **268**, 134–153.

Cockburn, B., and C.-W. Shu, 1998, The Runge–Kutta discontinuous Galerkin method for conservation laws V: Journal of Computational Physics, **141**, 199–224.

Crandall, M. G., and P. L. Lions, 1983, Viscosity solutions of Hamilton–Jacobi equations: Transactions of the American mathematical society, **277(1)**, 1–42.

———, 1984, Two approximations of solutions of Hamilton–Jacobi equations: Mathematics of com-

putation, **43(167)**, 1–19.

Cristiani, E., 2009, A fast marching method for Hamilton-Jacobi equations modeling monotone front propagations: Journal of Scientific Computing, **39**, 189–205.

Dellinger, J., and W. Symes, 1997, Anisotropic finite-difference traveltimes using a Hamilton–Jacobi solver: SEG Technical Program Expanded Abstracts, **16**, 1786–1789.

Dijkstra, E. W., 1959, A note on two problems in connection with graphs: Numerische Mathematik, **1**, 269–271.

Fomel, S., S. Luo, and H.-K. Zhao, 2009, Fast sweeping method for the factored eikonal equation: Journal of Computational Physics, **228**, 6440–6455.

Gray, S. H., and W. P. May, 1994, Kirchhoff migration using eikonal equation traveltimes: Geophysics, **54**, 810–817.

Gremaud, P. A., and C. M. Kuster, 2006, Computational study of fast methods for the Eikonal equation: SIAM Journal on Scientific Computing, **27**, 1803–1816.

Han, S., W. Zhang, and J. Zhang, 2017, Calculating qP-wave travel times in 2D TTI media by high-order fast sweeping methods with a numerical quartic equation solver: Geophysical Journal International, **210**, 1560–1569.

Hestholm, S., 1999, 3-D finite-difference visocelastic wave modeling including surface topography: Geophysical Journal International, 852–878.

Hole, D., and B. Zelt, 1995, 3-D finite difference reflection traveltimes: Geophysical Journal International, **121**, 427–434.

Improta, L., A. Zollo, A. Herrero, R. Frattini, J. Virieux, and P. DellÁversana, 2002, Seismic imaging of complex structures by non-linear traveltimes inversion of dense wide-angle data: application to a thrust belt: Geophysical Journal International, **151**, 264–278.

Jiang, G.-S., and D. Peng, 2000, Weighted ENO schemes for Hamilton–Jacobi equations: SIAM Journal on Scientific Computing, **21**, 2126–2143.

Jiang, G.-S., and C.-W. Shu, 1996, Efficient implementation of weighted eno schemes: Journal of Computational Physics, **126**, 202–228.

Jin, S., R. Madariaga, J. Virieux, and G. Lambaré, 1992, Two-dimensional asymptotic iterative elastic inversion: Geophysical Journal International, **108**, 575–588.

Kao, C. Y., S. Osher, and J. Qian, 2004, Lax-friedrichs sweeping schemes for static hamilton-jacobi equations: Journal of Computational Physics, **196**, 367–391.

Keller, J. B., 1962, A geometrical theory of diffraction: Journal of the Optical Society of America, **52**, 116–130.

Kim, S., 1999, On Eikonal solvers for anisotropic traveltimes: SEG Technical Program Expanded Abstracts 1999, 1875–1878.

Kim, S., and R. Cook, 1999, 3d traveltime computation using second order eno scheme: Geophysics, **64**, 1867–1876.

Lambaré, G., S. Operto, P. Podvin, P. Thierry, and M. Noble, 2003, 3-D ray+Born migration/inversion - part 1: theory: Geophysics, **68**, 1348–1356.

Le Bouteiller, P., M. Benjemaa, L. Métivier, and J. Virieux, 2018, An accurate discontinuous Galerkin method for solving point–source Eikonal equation in 2-D heterogeneous anisotropic media: Geophysical Journal International, **212**, 1498–1522.

Le Meur, H., 1994, Tomographie tridimensionnelle à partir des temps des premières arrivées des ondes P et S: PhD thesis, Université Paris VII.

Lelièvre, P. G., C. G. Farquharson, and C. A. Hurich, 2011, Inversion of first-arrival seismic traveltimes without rays, implemented on unstructured grids: Geophysical Journal International, **185**, 749–763.

Leung, S., and J. Qian, 2006, An adjoint state method for three-dimensional transmission traveltime tomography using first-arrivals: Communications in Mathematical Sciences, **4**, 249–266.

Li, F., C. W. Shu, Y. T. Zhang, and H. Zhao, 2008, A second order discontinuous galerkin fast sweeping method for Eikonal equations: Journal of Computational Physics, **227**, 8191–8208.

Liu, X., S. Osher, and T. Chan, 1994, Weighted essentially non-oscillatory schemes: Journal of Computational Physics, **115**, 200–212.

Luo, S., and J. Qian, 2011, Factored singularities and high-order Lax-Friedrichs sweeping schemes

for point-source traveltimes and amplitudes: Journal of Computational Physics, **230**, 4742–4755.

Luo, S., and H. Zhao, 2016, Convergence analysis of the fast sweeping method for static convex Hamilton–Jacobi equations: Research in the Mathematical Sciences, **3**, 35.

Mendes, M., 2000, Green's function interpolations for prestack imaging: Geophysical Prospecting, **48**, 49–62.

Mirebeau, J.-M., 2014, Anisotropic fast-marching on cartesian grids using lattice basis reduction: SIAM Journal on Numerical Analysis, **52**, 1573–1599.

Mirebeau, J.-M., and J. Portegies, 2018, Hamiltonian fast marching: A numerical solver for anisotropic and non-holonomic eikonal pdes. Preprint.

Noble, M., A. Gesret, and N. Belayouni, 2014, Accurate 3-d finite difference computation of travel time in strongly hetrogeneous media: Geophysical Journal International, **199**, 1572–1585.

Operto, S., G. Lambaré, P. Podvin, and P. Thierry, 2003, 3-D ray-Born migration/inversion. part 2: application to the SEG/EAGE overthrust experiment: Geophysics, **68**, 1357–1370.

Osher, S., 1993, A level set formulation for the solution of the dirichlet problem for Hamilton–Jacobi equations: SIAM Journal on Mathematical Analysis, **24**, 1145–1152.

Pica, A., 1997, Fast and accurate finite-difference solutions of the 3-D eikonal equation parameterized in celerity: 67^{th} Annual SEG Meeting and Exposition, Expanded Abstracts, Society of Exploration Geophysics, 1774–1777.

Podvin, P., and I. Lecomte, 1991, Finite difference computation of traveltimes in very contrasted velocity model : a massively parallel approach and its associated tools: Geophysical Journal International, **105**, 271–284.

Qian, J., W. W. Symes, and J. A. Dellinger, 2001, A full-aperture anisotropic Eikonal solver for quasi-P traveltimes: SEG Technical Program Expanded Abstracts 2001, 129–132.

Qian, J., Y.-T. Zhang, and H.-K. Zhao, 2007a, A fast sweeping method for static convex Hamilton–Jacobi equations: Journal of Scientific Computing, **31**, 237–271.

———, 2007b, Fast sweeping methods for Eikonal equations on triangular meshes: SIAM Journal on Numerical Analysis, **45**, 83–107.

Qin, F., Y. Luo, K. B. Olsen, W. Cai, and G. T. Schuster, 1992, Finite difference solution of the eikonal equation along expanding wavefronts: Geophysics, **57**, 478–487.

Regone, C., J. Stefani, P. Wang, C. Gerea, G. Gonzalez, and M. Oristaglio, 2017, Geologic model building in SEAM Phase II - Land seismic challenges: The Leading Edge, **36**, 738–749.

Runborg, O., 2007, Mathematical models and numerical methods for high frequency waves: Communications in Computational Physics, **2**, 827–880.

Sethian, J. A., 1996, A fast marching level set method for monotonically advancing fronts: Proceedings of the National Academy of Sciences of the United States of America, **93**, 1591–1595.

———, 1999, Fast marching methods: SIAM Review, **41**, 199–235.

Shu, C.-W., and S. Osher, 1988, Efficent implementation of essentially non-oscillatory shock-capturing schemes: Journal of Computational Physics, **77**, 439–471.

———, 1989, Efficent implementation of essentially non-oscillatory shock-capturing schemes, ii.: Journal of Computational Physics, **83**, 32–78.

Slawinski, M. A., 2003, Seismic waves and rays in elastic media: Elsevier Science.

Taillandier, C., M. Noble, H. Chauris, and H. Calandra, 2009, First-arrival travel time tomography based on the adjoint state method: Geophysics, **74**, no. 6, WCB1–WCB10.

Tarrass, I., L. Giraud, and P. Thore, 2011, New curvilinear scheme for elastic wave propagation in presence of curved topography: Geophysical Prospecting, **59**, 889–906.

Tavakoli F., B., S. Operto, A. Ribodetti, and J. Virieux, 2017, Slope tomography based on eikonal solvers and the adjoint-state method: Geophysical Journal International, **209(3)**, 1629–1647.

Tavakoli F., B., A. Ribodetti, J. Virieux, and S. Operto, 2015, An iterative factored eikonal solver for TTI media: Expanded Abstracts, 85^{th} Annual SEG Meeting (New Orleans), SEG technical program expanded abstracts 2015, 3576–3581.

Thomsen, L. A., 1986, Weak elastic anisotropy: Geophysics, **51**, 1954–1966.

Trinh, P. T., R. Brossier, L. Métivier, L. Tavard, and J. Virieux, 2017, Efficient 3D elastic FWI using a spectral-element method: 87th SEG Conference and Exhibition 2017, Houston, 1533–1538.

———, 2018, Efficient 3D time-domain elastic and viscoelastic Full Waveform Inversion using a

spectral-element method on flexible Cartesian-based mesh: Geophysics.

Tsai, Y.-H. R., L.-T. Chen, S. Osher, and H.-K. Zhao, 2003, Fast sweeping algorithms for a class of Hamilton–Jacobi equations: SIAM Journal on Numerical Analysis, **41**, 673–694.

Tsitsiklis, J. N., 1995, Efficient algorithms for globally optimal trajectories: IEEE Transactions on Automatic Control, **40**, 1528–1538.

Tsvankin, I., 1997, Anisotropic parameters and P-wave velocity for orthorhombic media: Geophysics, **62**, 1292–1309.

van Trier, J., and W. W. Symes, 1991, Upwind finite-difference calculation of traveltimes: Geophysics, **56**, 812–821.

Vanelle, C., and D. Gajewski, 2002, Second-order interpolation of traveltimes: Geophysical Prospecting, **50**, 73–83.

Vidale, D., 1988, Finite-difference calculation of travel time: Bulletin of the Seismological Society of America, **78**, 2062–2076.

Waheed, U., and T. Alkhalifah, 2017, A fast sweeping algorithm for accurate solution of the tilted transversely isotropic eikonal equation using factorization: Geophysics, **82**, no. 6, WB1–WB8.

Waheed, U. B., C. E. Yarman, and G. Flagg, 2015, An iterative, fast-sweeping-based eikonal solver for 3D tilted anisotropic media: Geophysics, **80**, no. 3, C49–C58.

Zhang, L., J. W. Rector III, and G. M. Hoversten, 2005a, Eikonal solver in the celerity domain: Geophysical Journal International, **162**, 1–8.

Zhang, Y., H. Zhao, and S. Chen, 2005b, Fixed–point iterative sweeping methods for static Hamilton–Jacobi equations: Methods and Applications of Analysis, **13**, 299–320.

Zhang, Y.-T., S. Chen, F. Li, H. Zhao, and C.-W. Shu, 2011, Uniformly accurate discontinuous Galerkin fast sweeping methods for Eikonal equations: SIAM Journal on Scientific Computing, **33**, 1873 – 1896.

Zhao, H., 2005, A fast sweeping method for eikonal equations: Mathematics of computation, **74**, 603–627.

Zienkewicz, O. C., and K. Morgan, 1983, Finite elements and approximation: J. Wiley and Sons.